MA 202 - Mathematics IV

Numerical Analysis of 3 Body Problem Restricted to 2 Dimensions

Viramgami Gaurav - 19110106 | Yash More - 19110123 | Kritika Kumawat - 19110185 | Pahuni Jain - 19110021 | Maddela Siddarth - 18110097

Driver Code Results File

0.0.1 To center align all the output figures

[1]:
```
from IPython.core.display import HTML as Center

Center(""" <style>
.output_png {
    display: table-cell;
    text-align: center;
    vertical-align: middle;
}
</style> """)
```

[1]:
```
<IPython.core.display.HTML object>
```

0.0.2 Importing Files conatining classes and constants

`Important_classes.py`: Contains defined classes including `Object2D`, `TwoBodySystem`, `ThreeBodySystem`, and `NBodySystem` `Global_constants.py`: Contains constants and parameters used in the code

[2]:
```
from Important_classes import *
from Global_constants import *
```

```
pygame 2.0.1 (SDL 2.0.14, Python 3.8.2)
Hello from the pygame community. https://www.pygame.org/contribute.html
```

Since, the celectial orbits preserve energy, we have to use Symplectic methods to numerically solve the trajectories of such objects. We have majorly discussed three methods: - Euler Method - `Non Energy Preserving` - Euler Cromer Method - `Energy Preserving` - Velocity Verlet Method - `Energy Preserving`

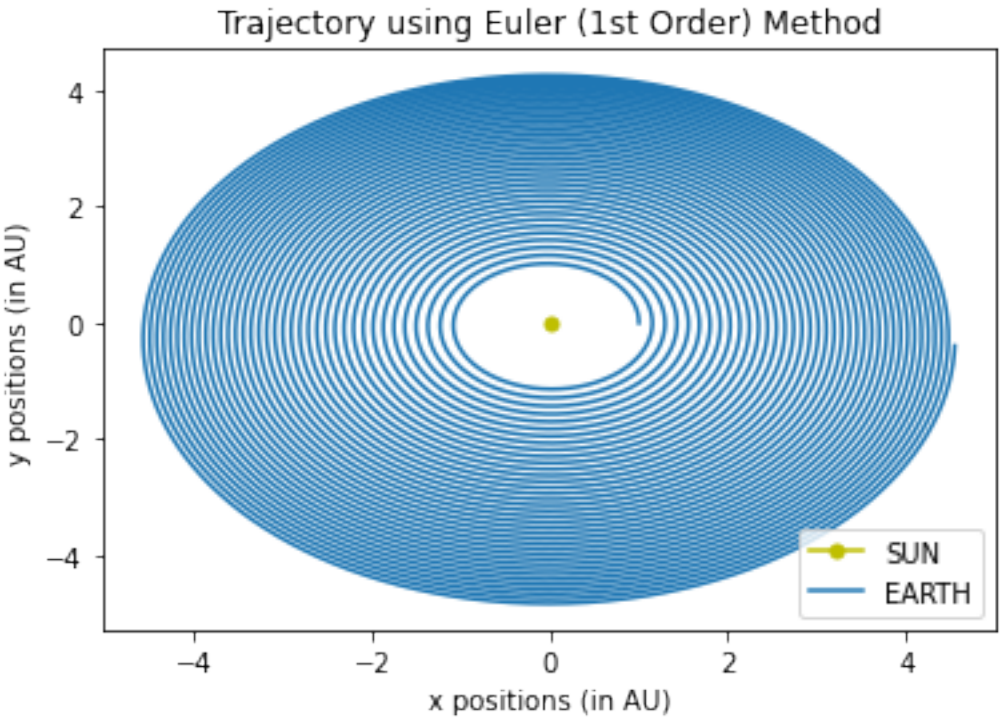
Case: 1

2 Body System: Sun - Earth System

[3]:
```
stepsize = 0.002
num_iterations = 100000
case1 = TwoBodySystem(OBJECTS["SUN"], OBJECTS["EARTH"])
```

Non Energy Preserving Euler Method

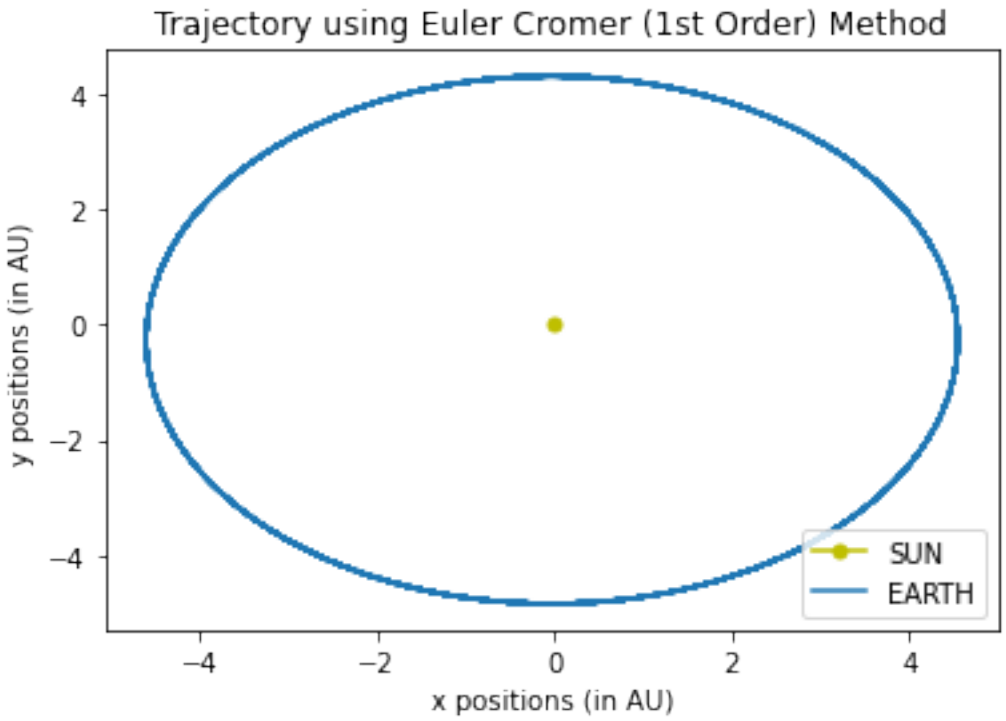
[4]:
```
case1.euler_method(stepsize, num_iterations)
case1.plot_euler_trajectory()
```

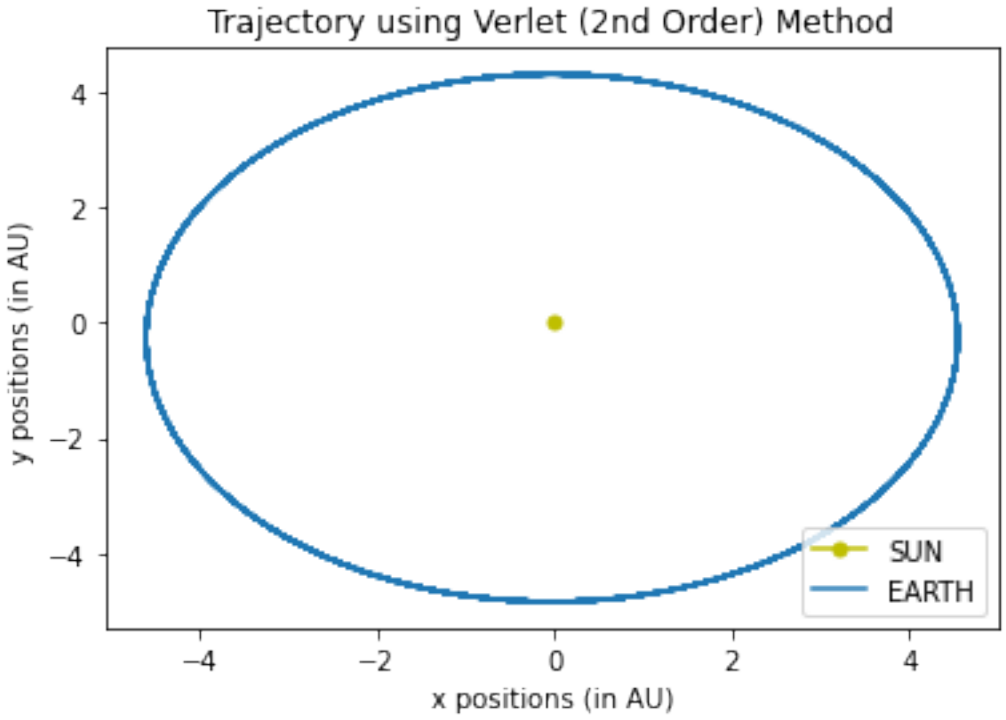


We can observe that the object is orbiting in a spiral path since its energy is not conserved and it ends up colliding with the central object. To avoid such situation we need to use Sympletic methods to solve the numerical problems which involves Energey Preserving Nature. The following two methods: `Euler Cromer Method` and `Verlet Method` are Energy Preserving Methods, thus giving stable orbit.

[5]:
```
case1.euler_cromer_method(stepsize, num_iterations)
case1.plot_euler_cromer_trajectory()

case1.verlet_method(stepsize, num_iterations)
case1.plot_verlet_trajectory()
```


Trajectory using Euler Cromer (1st Order) Method
y positions (in AU)
x positions (in AU)
SUN
EARTH
4
2
0
−2
−4
−4
−2
0
2
4


Trajectory using Verlet (2nd Order) Method
y positions (in AU)
x positions (in AU)
SUN
EARTH
4
2
0
−2
−4
−4
−2
0
2
4

Case: 2

3 Body System: Sun - Mars - Jupiter System

[4]:
```
stepsize = 0.002
num_iterations = 100000
case2 = ThreeBodySystem(OBJECTS["SUN"], OBJECTS["MARS"], OBJECTS["JUPITER"])
```

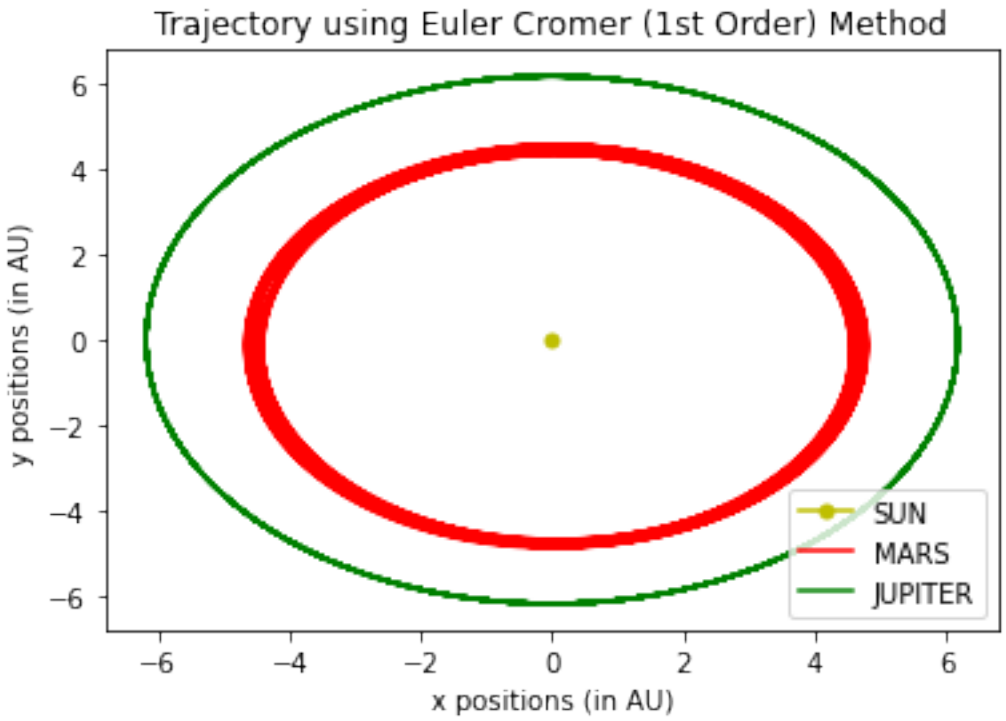
Non Energy Preserving Euler Method

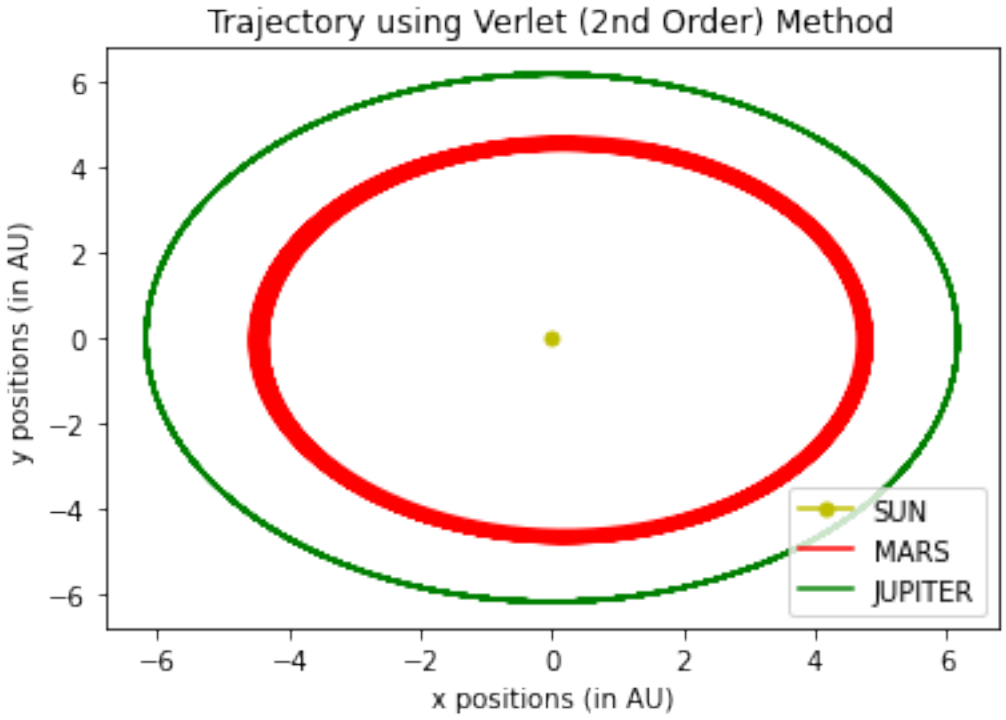
[5]:
```
case2.euler_method(stepsize, num_iterations)
case2.plot_euler_trajectory()
```

The following two methods: `Euler Cromer Method` and `Verlet Method` are Energy Preserving Methods, thus giving stable orbit.

[6]:
```
case2.euler_cromer_method(stepsize, num_iterations)
case2.plot_euler_cromer_trajectory()

case2.verlet_method(stepsize, num_iterations)
case2.plot_verlet_trajectory()
```


Trajectory using Euler Cromer (1st Order) Method
y positions (in AU)
x positions (in AU)
SUN
MARS
JUPITER


Trajectory using Verlet (2nd Order) Method
y positions (in AU)
x positions (in AU)
SUN
MARS
JUPITER

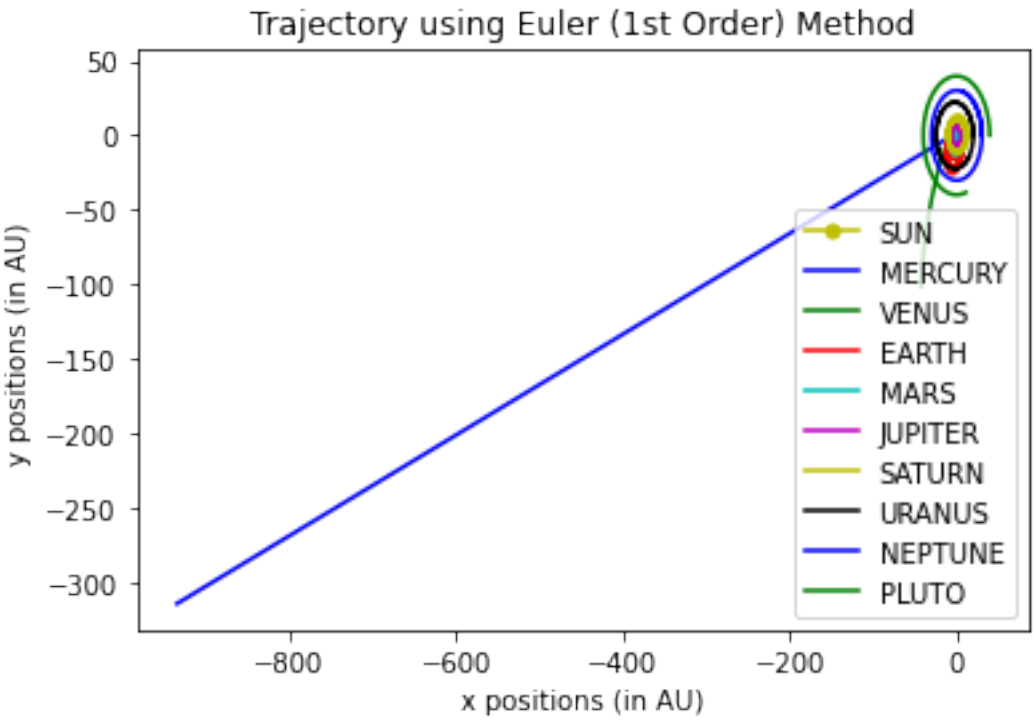
Case: 3

N Body System: Solar System

[3]:
```
stepsize = 0.02
num_iterations = 10000
case3 = NBodySystem(OBJECTS["SUN"], [OBJECTS["MERCURY"], OBJECTS["VENUS"],␣
↪OBJECTS["EARTH"], OBJECTS["MARS"], OBJECTS["JUPITER"], OBJECTS["SATURN"],␣
↪OBJECTS["URANUS"], OBJECTS["NEPTUNE"], OBJECTS["PLUTO"]])
```

Non Energy Preserving Euler Method

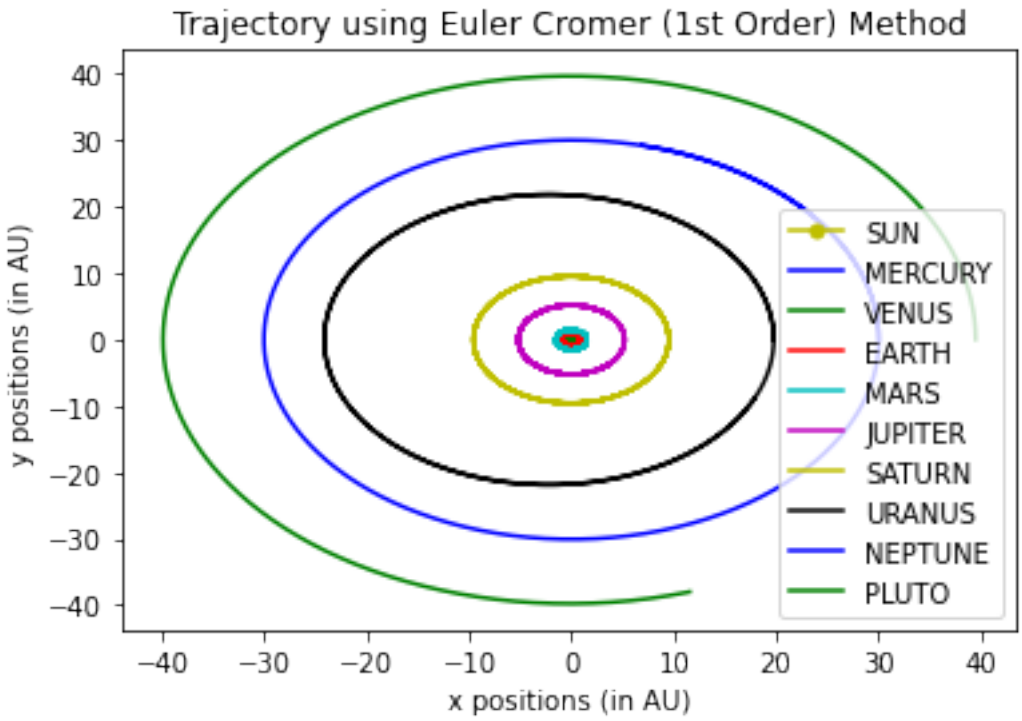
[4]:
```
case3.euler_method(stepsize, num_iterations)
case3.plot_euler_trajectory()
```

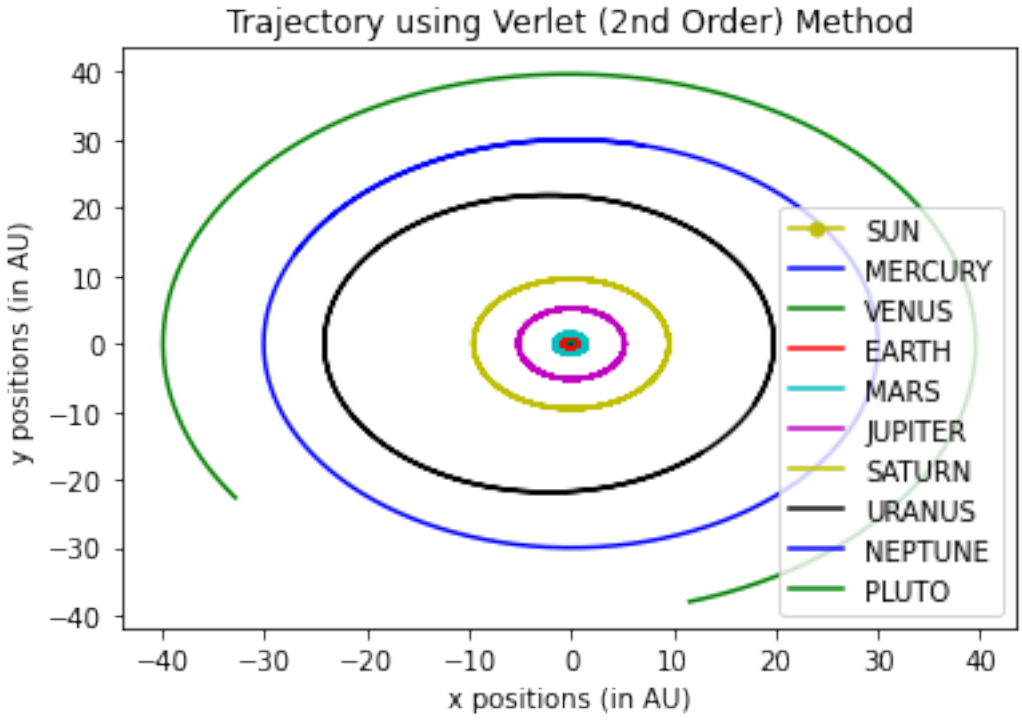


The following two methods: `Euler Cromer Method` and `Verlet Method` are Energy Preserving Methods, thus giving stable orbit.

[4]:
```
case3.euler_cromer_method(stepsize, num_iterations)
case3.plot_euler_cromer_trajectory()

case3.verlet_method(stepsize, num_iterations)
case3.plot_verlet_trajectory()
```


Trajectory using Euler Cromer (1st Order) Method
y positions (in AU)
x positions (in AU)
SUN
MERCURY
VENUS
EARTH
MARS
JUPITER
SATURN
URANUS
NEPTUNE
PLUTO


Trajectory using Verlet (2nd Order) Method
y positions (in AU)
x positions (in AU)
SUN
MERCURY
VENUS
EARTH
MARS
JUPITER
SATURN
URANUS
NEPTUNE
PLUTO

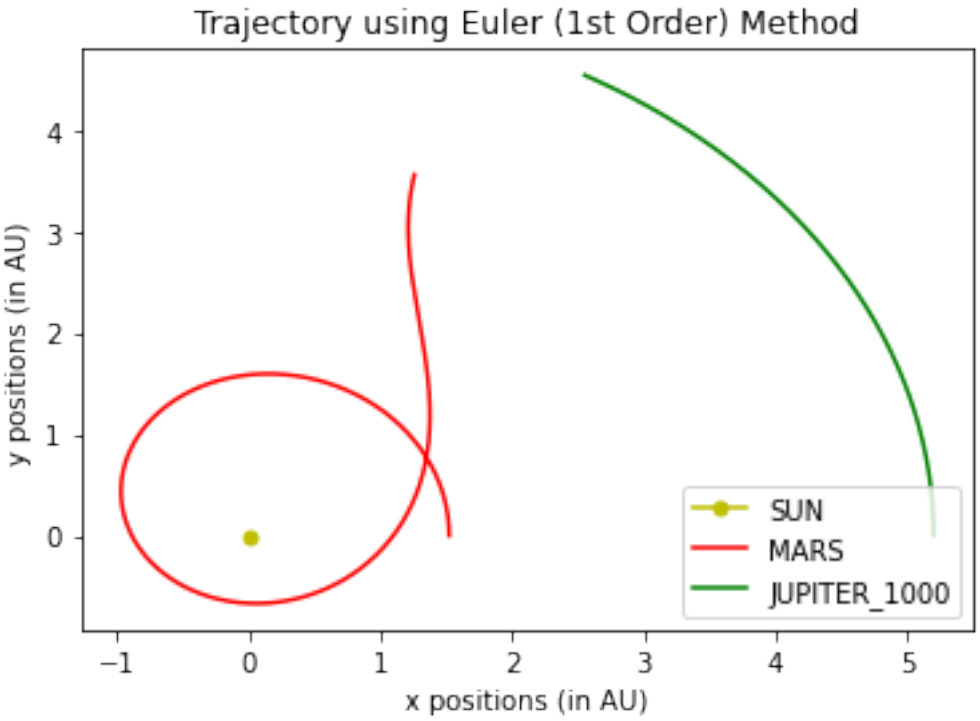
Miscellaneous Cases

Increasing Mass Of Jupiter by 1000 times in Case 2

[3]:
```
stepsize = 0.002
num_iterations = 1000
case4 = ThreeBodySystem(OBJECTS["SUN"], OBJECTS["MARS"], OBJECTS["JUPITER_1000"])
```

Non Energy Preserving Euler Method

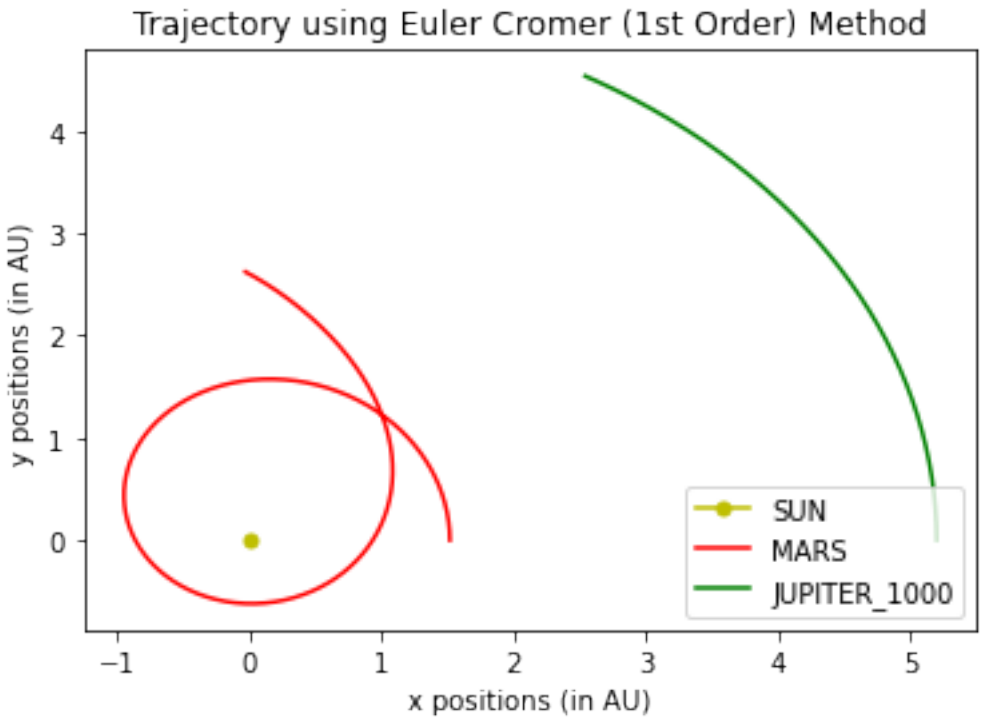
[4]:
```
case4.euler_method(stepsize, num_iterations)
case4.plot_euler_trajectory()
```

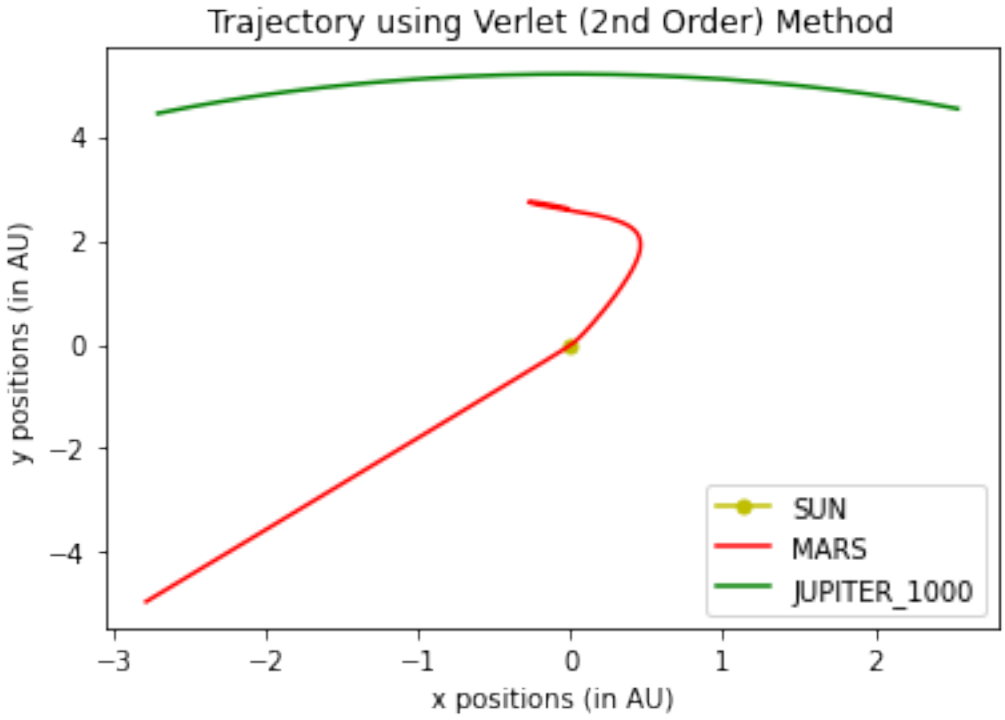


The following two methods: `Euler Cromer Method` and `Verlet Method` are Energy Preserving Methods, thus giving stable orbit.

[4]:
```
case4.euler_cromer_method(stepsize, num_iterations)
case4.plot_euler_cromer_trajectory()

case4.verlet_method(stepsize, num_iterations)
case4.plot_verlet_trajectory()
```


Trajectory using Euler Cromer (1st Order) Method
y positions (in AU)
x positions (in AU)
SUN
MARS
JUPITER_1000

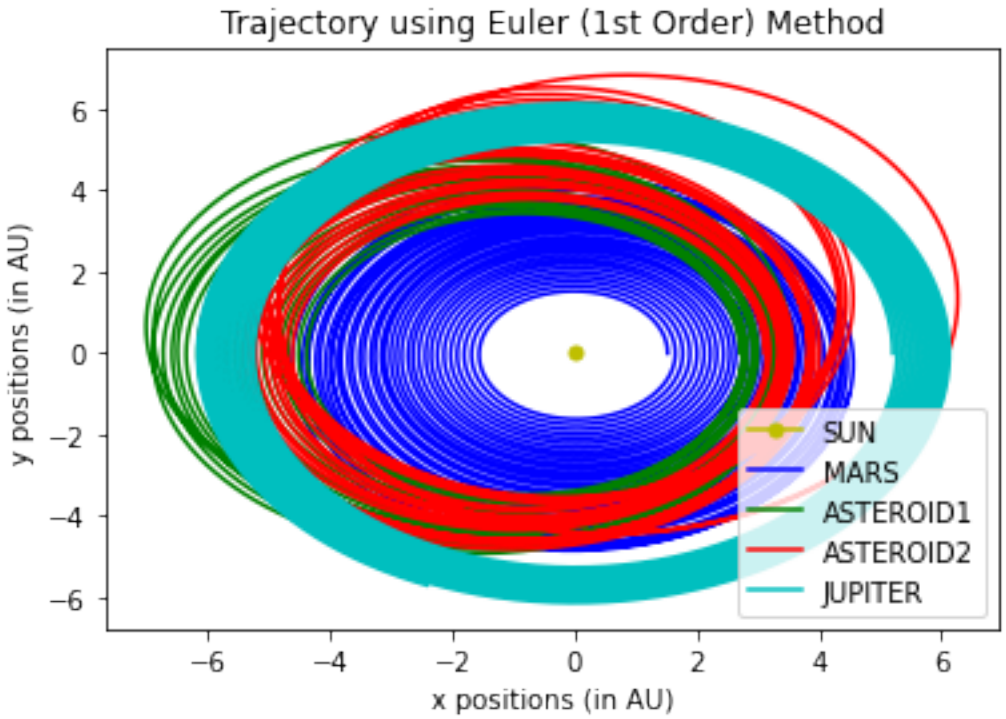

Trajectory using Verlet (2nd Order) Method
y positions (in AU)
x positions (in AU)
SUN
MARS
JUPITER_1000

Asteroid Belt between Mars and Jupiter

[3]:
```
stepsize = 0.002
num_iterations = 100000
case5 = NBodySystem(OBJECTS["SUN"], [OBJECTS["MARS"], OBJECTS["ASTEROID1"],␣
↪OBJECTS["ASTEROID2"], OBJECTS["JUPITER"]])
```

Non Energy Preserving Euler Method

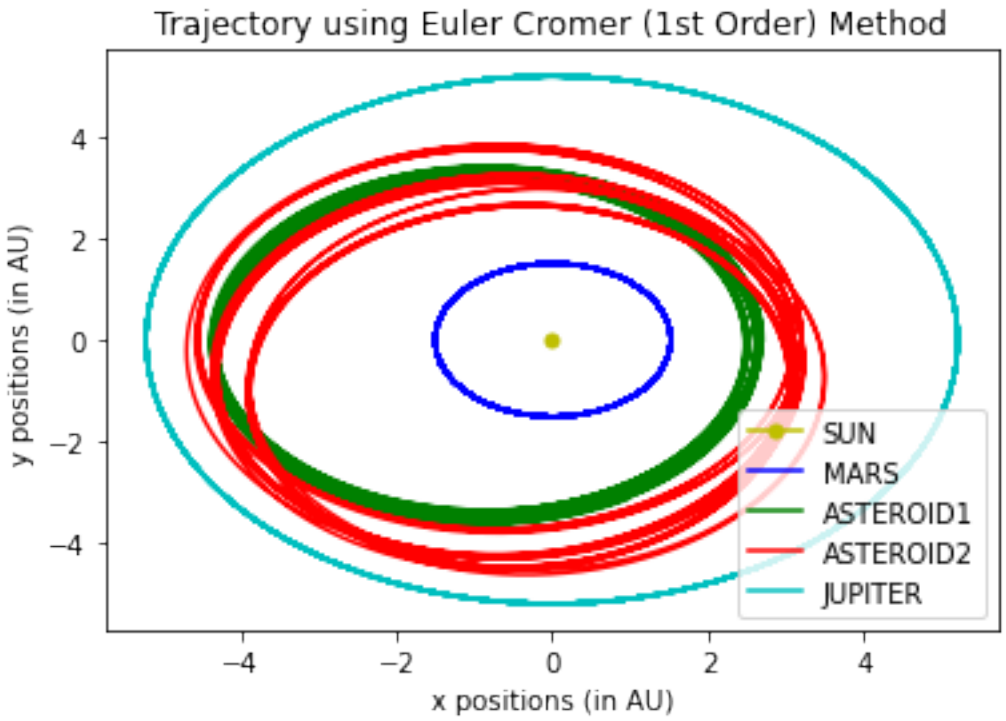
[4]:
```
case5.euler_method(stepsize, num_iterations)
case5.plot_euler_trajectory()
```

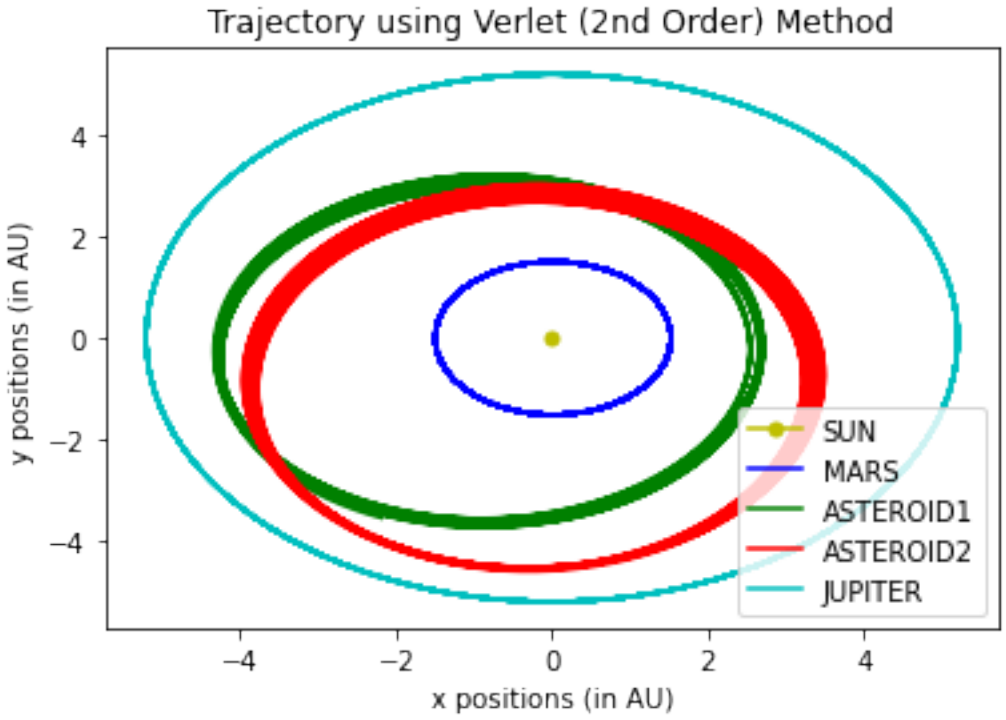


The following two methods: `Euler Cromer Method` and `Verlet Method` are Energy Preserving Methods, thus giving stable orbit.

[4]:
```
case5.euler_cromer_method(stepsize, num_iterations)
case5.plot_euler_cromer_trajectory()

case5.verlet_method(stepsize, num_iterations)
case5.plot_verlet_trajectory()
```


Trajectory using Euler Cromer (1st Order) Method
y positions (in AU)
x positions (in AU)
SUN
MARS
ASTEROID1
ASTEROID2
JUPITER


Trajectory using Verlet (2nd Order) Method
y positions (in AU)
x positions (in AU)
SUN
MARS
ASTEROID1
ASTEROID2
JUPITER